Meg – Hacker Tools: Endpoint scan the masses! □□

BY ANNA HAMMOND · FEBRUARY 1, 2022 · LAST UPDATED ON AUGUST 8, 2026

Want to read the latest version? Visit the link below:
https://www.intigriti.com/researchers/blog/hacking-tools/hacker-tools-meg

You've just enumerated all the subdomains of your target and what? There's 400 of them? Are you going to start individual scans to find endpoints on them? No you're not! You're going to use Meg, of course!

□□ What is Meg?

Meg is not the girl nextdoor, no it's an amazing tool you need to know about! As more and more people get into bug bounty, more and more scans are being launched and some servers are just not ready for them. We want to scan a ton of endpoints, but we want the server to stay responsive, so that we can continue scanning.

Tomnomnom has come up with the brilliant idea of not just scanning our targets one-by-one, but distributing traffic between them. Instead of barraging a single target with 100 requests per second, we send 1 request to 100 targets per second. This way, everyone can keep up and we still receive the same amount of results!

□□ Installing Meg

Installing Meg couldn't be easier! Just follow these steps

- Go to the releases page on GitHub
- Download the binary for your architecture
- Run `tar -zxvf binary.tgz` to extract it
- Enjoy!

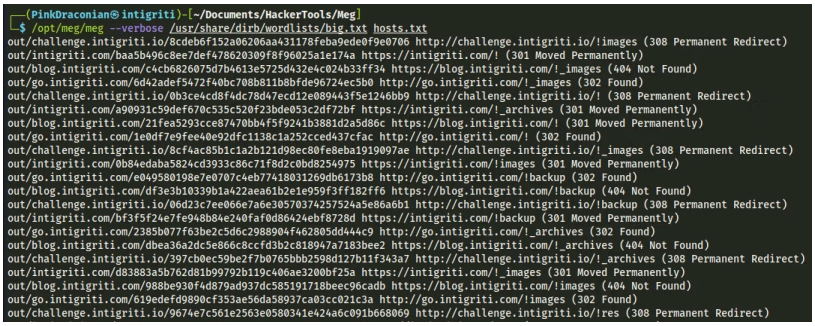
□□ Our first run!

Let's get into it! Let's finally start using Meg!

The things you need in order to run Meg are:

- A wordlist of endpoints you want to scan. SecLists is a great place to get started.
- A list of targets to scan. In this case, I'll take some intigriti subdomains, however you will probably want to take the output from another discovery tool!

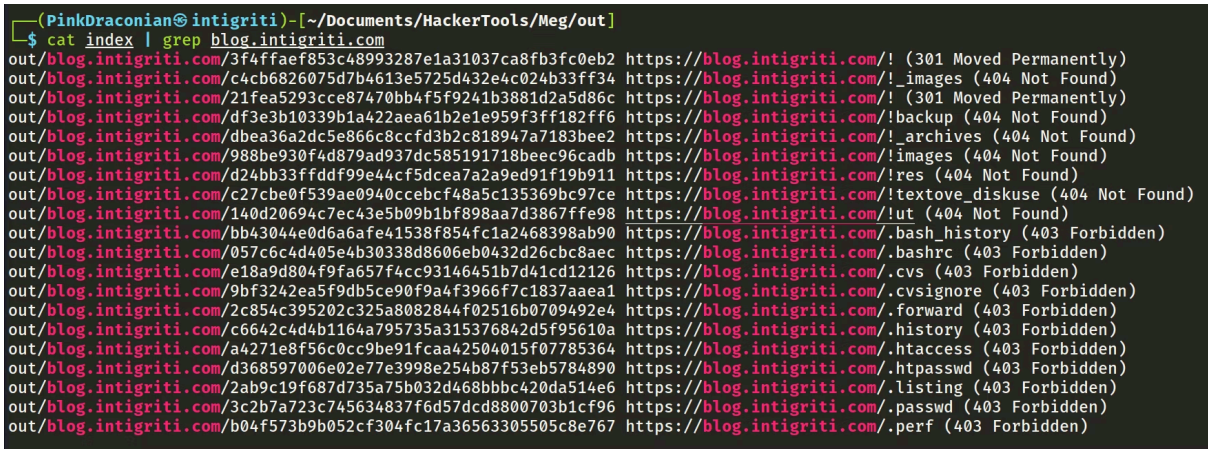
Now we're really ready to go! I'll run `meg --verbose wordlist hosts.txt` Note that I'm using the `--verbose` flag here just to make what the tool is doing visible. This is of course not needed. Let's assess the command line output of the tool!



Meg output

The output here shows the location where it has stored the result of the request, then the endpoint and then the status code. We can see here how it sends the same request to each endpoint and gathers it's response. This is how the load for the server is greatly decreased!

Now, without the `--verbose` option, the tool doesn't print out this information, but it saves it to `out/index` (output directory can be changed). From there you can simply use grep to get all the results for specific targets as shown here



Meg index grepping

You can also see the responses for these requests by just looking in the files for them, an example of which is shown below

```
┌──(PinkDraconian㉿intigriti)-[~/Documents/HackerTools/Meg]
└─$ cat out/blog.intigriti.com/b04f573b9b052cf304fc17a36563305505c8e767
https://blog.intigriti.com/.perf

> GET /.perf HTTP/1.1
> Host: blog.intigriti.com
> User-Agent: Mozilla/5.0 (compatible; meg/0.2; +https://github.com/tomnomnom/meg)

< HTTP/1.1 403 Forbidden
< Content-Length: 146
< Strict-Transport-Security: max-age=31536000
< X-Ac: 1.ams _atomic_ams
< Server: nginx
< Date: Wed, 26 Jan 2022 10:00:39 GMT
< Content-Type: text/html

<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
<hr><center>nginx</center>
</body>
</html>
```

Meg showing output file

Features

You may have seen some things you don't really fancy in the results shown before. Perhaps you only want to capture 200 OK responses or you want to send HEAD requests instead of GET requests. All of that is of course possible. Let's discuss Meg's `--help` page!

```
┌──(PinkDraconian㉿intigriti)-[~/Documents/HackerTools/Meg]
└─$ /opt/meg/meg --help
Request many paths for many hosts

Usage:
  meg [path|pathsFile] [hostsFile] [outputDir]

Options:
  -c, --concurrency <val>    Set the concurrency level (defaut: 20)
  -d, --delay <millis>       Milliseconds between requests to the same host (defaut: 5000)
  -H, --header <header>      Send a custom HTTP header
  -L, --location             Follow redirects / location header
  -r, --rawhttp              Use the rawhttp library for requests (experimental)
  -s, --savestatus <status>  Save only responses with specific status code
  -t, --timeout <millis>     Set the HTTP timeout (default: 10000)
  -v, --verbose              Verbose mode
  -X, --method <method>      HTTP method (default: GET)

Defaults:
  pathsFile: ./paths
  hostsFile: ./hosts
  outputDir:  ./out

Paths file format:
  /robots.txt
  /package.json
  /security.txt

Hosts file format:
  http://example.com
  https://example.edu
  https://example.net

Examples:
  meg /robots.txt
  meg paths.txt hosts.txt output
```

Meg –help

- `-c` or `--concurrency` : This option can be used to change the concurrency level. Higher numbers here mean more requests being sent out at once.
- `-d` or `--delay` : This option relates to the delay between every request to the same host in milliseconds. This tool is made to be slower to the server, however, in certain configurations, it can still be too fast. Luckily the default here is already 5 seconds.

- `-H` or `--header` : Need to set a specific header for each request, worry no more and set this option!
- `-L` or `--location` : Getting a lot of 302's? Use this option to follow those redirects!
- `-s` or `--savestatus` : Only save responses with specific status codes.
- `-t` or `--timeout` : Failing requests will timeout after this amount of milliseconds. The default is 10000.
- `-v` or `--verbose` : If you need some more output, then this option is for you!
- `-X` or `--method` : Change the request method from GET to for example HEAD. I highly recommend playing around with sending HEAD requests!

Conclusion

Meg is a simple, yet helpful tool designed to help you get more efficient and to go easier on your targets! Start using it today to hack even more efficiently!

If you would like to recommend a tool for us to cover next week, then be sure to let us know down below. Also be sure to check out all the previous Hacker Tools articles, such as the last one on EyeWitness.

Did you know that there is a video accompanying this article? Check out the playlist!